Using deep autoencoders to investigate image matching in visual navigation

Christopher Walker[1], Paul Graham[2], Andrew Philippides[1]

[1] Centre for Computational Neuroscience and Robotics, Department of Informatics, University of Sussex, Brighton, UK
{chris.walker, andrewop}@sussex.ac.uk

[2] Centre for Computational Neuroscience and Robotics, School of Life Sciences, University of Sussex, Brighton, UK
p.r.graham@sussex.ac.uk

Abstract. This paper discusses the use of deep autoencoder networks to find a compressed representation of an image, which can be used for visual navigation. Images reconstructed from the compressed representation are tested to see if they retain enough information to be used as a visual compass (in which an image is matched with another to recall a bearing/movement direction) as this ability is at the heart of a visual route navigation algorithm. We show that both reconstructed images and compressed representations from different layers of the autoencoder can be used in this way, suggesting that a compact image code is sufficient for visual navigation and that deep networks hold promise for finding optimal visual encodings for this task.

Keywords: Visual navigation, insect-inspired robotics, deep neural network, autoencoder

1 Introduction

Navigation is an important ability for both natural and artificial agents [1]. When looking to nature for inspiration, engineers have turned to ants and bees as they use vision to navigate long distances through complex natural habitats despite limited neural and sensory resources [2-5]. They achieve this task by using retinotopic image-matching methods, which has inspired a range of bio-inspired algorithms (Ants: [6-8]; Bees: [3]; Review: [9]). We have previously shown that panoramic images can be used for navigation in desert ant-inspired algorithms even if images are low-resolution [10], processed through coarse visual filters modelled on parts of the drosophila visual system [11], or processed so that only the height of objects against the skyline is used [12]. This work not only demonstrates the robustness of using low-resolution images for navigation but also that they can be better than high-resolution images [10]. However, while we know that desert ants have low-resolution vision, we do not know how

they encode images so that they are best-suited for navigation, nor do we know what visual encodings would be optimal for a navigating agent. In this paper, we investigate this question by examining the compressed visual encodings that arise from deep autoencoder networks trained on natural images. As autoencoders automatically derive low dimensional representations of the underlying data in an unsupervised manner, they are a relatively assumption-free methodology with which to investigate optimal visual encodings while also shedding light on insect visual systems.

The opportunity to use these methods arises because desert ant foragers are task specialists whose sole goal is to visually navigate between nest and food. We can therefore assume that their visual system has been honed by evolution for this task. Thus, we can use AI methodologies as statistical engines to investigate the optimal encodings for navigating with image matching. One such method is to use an autoencoder. Autoencoders are neural networks which are trained to reconstruct their input at the output and have a single hidden layer, usually much smaller in size than the input and output layers, which forces the network to learn a compressed representation of the input. Because these encodings represent statistical regularities from the visual world, they can be used to explore the visual computations that evolution might also have discovered allowing us to hypothesise about how an insect's visual pathway might process images.

A

B

Fig. 1. Unwrapped panoramic image collected from route on University of Sussex campus. First image is colour with dimensions of 360 x 90 pixels, as extracted from video. Second image is scaled monochrome with dimensions of 180 x 45 pixels, as used in all experiments.

Specifically, here we use deep autoencoder networks trained with images gathered by a robot equipped with a panoramic video camera navigating through a wooded environment. A deep autoencoder is an autoencoder with more than one hidden layer, which is particularly useful for this kind of task, as previous work has shown that different layers of the networks extract task-relevant features at different levels of abstraction [13]. As our task is navigation, we examine the encodings produced by different layers of an autoencoder network, to assess how well their output can be used to regain a bearing from a memorised image. This simple verification shows that the information needed for visual route navigation is retained in the encoding as this ability lies at the heart of our route navigation algorithms [7]. As a corollary, we can also examine if there is a compact encoding suited to robotic route navigation with our algorithm. While this work will thus aid robotic navigation, more importantly, it is a first step towards using deep learning to understand insect visual encodings.

2 Methods

2.1 Panoramic images

The images used in this experiment are collected from a Unibot robot built by Creative Robotics Ltd (http://www.creative-robotics.com/?q=unibot) using a Kodak Pixpro SP360 panoramic camera fixed to the top of the robot. Video footage was recorded as the robot was driven along two different routes through wooded land on the University of Sussex campus at Falmer (sample video can be seen at: https://www.youtube.com/watch?v=f9fkPABQOhg). The video is unwrapped using the Pixpro SP360 desktop software so that the entire panorama is seen as a wide strip, where the forward direction of travel is always in the centre of the image and the far left and right edges are the view behind the robot. Individual frames are then extracted from the video footage which was recorded at 30 frames per second. The ffmpeg library is used with the default bicubic scaling algorithm to extract each frame and scale the output images to 180 x 45 pixels in monochrome, to reduce the complexity of the input to the network. Sample images can be seen in Fig. 1

2.2 Assessing navigational information via the image difference function (IDF)

In our route navigation algorithms, we compare the current perceived view with remembered views with the objective of choosing the best direction to move next. Thus, to be useful for familiarity-based route navigation, images must be processed in such a way that we can reliably find the heading at which the current image best matches a stored view. A method which can assess whether processed images retain this property, and which is agnostic of the details of the route navigation algorithm, is the rotational image difference function (RIDF) [14]. The RIDF is based on the image difference function (IDF) a pixel-wise difference between two images X and Y defined as:

$$IDF(X,Y) = \frac{1}{P}\sum_i \sum_j |X(i,j) - Y(i,j)| \quad (1)$$

where X(i,j) is the pixel in the i'th row and j'th column of image X and P is the number of pixels. The more similar the two images, the lower the IDF value will be. Here we are using the absolute pixel difference instead of the r.m.s. pixel difference originally used in [14].

In the above, X and Y are assumed to be aligned to a common heading. To get the RIDF, we rotate one of the images through 360° and find the minimum IDF value across all rotations. In this way we find the heading at which the current image best-matches the remembered view. If the images are sufficiently near each other and the encoding has persevered navigationally useful information, the best-matching heading of the current image will be similar to the heading of the remembered image and the RIDF has a characteristic V-shape around this minimum value [14,15,12]. To assess whether sufficient retinotopic information is retained in images encoded by deep networks, we thus compare images with rotated versions of themselves as the presence of the characteristic V is indicative of the presence of homing information.

2.3 Deep autoencoder networks

Autoencoder networks are a general class of network which try to produce as output a reconstruction of their input and are often used for dimensionality reduction by passing the input through a hidden layer which has a lower dimension than the input/output layers. Deep neural networks, which are defined as having more than one hidden layer, are suitable for use with the autoencoder technique as their multi-layered structure allows features of increasing complexity to be learned at different layers. Equally the autoencoder is suited to the deep network structure – which requires a great deal of training data – as the training data in this case is self-labelled as the autoencoder is trained to minimise the difference between the output and input.
Such *deep autoencoder networks* (Fig. 2) can thus be used to identify features of datasets with high-dimensionality, such that the data can be represented in a compressed form as low-dimensional codes [16,17].

The autoencoder is formed of two distinct parts, an encoder which transforms the high-dimensional data into a low-dimensional code, and a decoder which reconstructs the original data from the code (Fig. 2). The code layer, which connects the encoder and decoder, has fewer features than the input creating a bottleneck. This forces the encoder to find a compact representation of the input data that the decoder can use to reconstruct the original input as accurately as possible. Here we use a fully connected autoencoder with an input image size of 180x45 pixels. The image is converted into a one-dimensional array, giving 8100 features at the input layer. The encoder has 5 hidden layers of decreasing size (4096, 2048, 512, 128, 64) with the decoder having the inverse. The layer sizes were chosen arbitrarily to reduce the network to 64 features at the narrowest layer. Networks were constructed and trained in the Tensorflow

software (https://www.tensorflow.org/) using mini-batch gradient descent with an initial learning rate of 0.001 and the Adam optimiser [18] to speed up convergence and reduce overfitting.

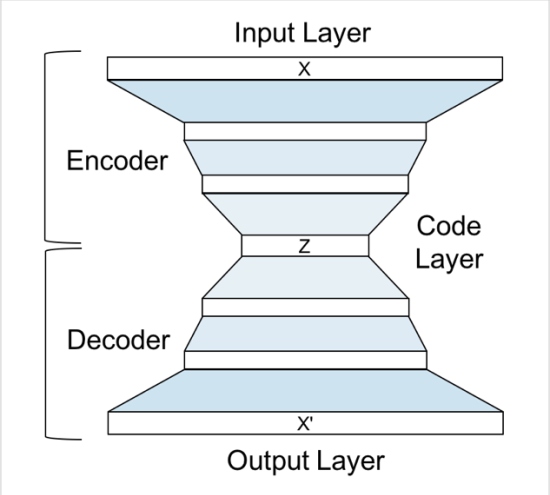

Fig. 2. Structure of a deep autoencoder network with fully connected hidden layers. The structure of the decoder is a mirror image of the encoder and each pair of equivalent layers share the same weights. An original image X is given as input and the network is trained to reconstruct X as X′ at the output layer. The narrow code layer at the centre forces the network to create a lower-dimensional encoding Z of X, from which it is able to reconstruct X'.

The dataset contained 2707 images in a randomised order. The images were normalised and the first 100 were set aside for testing; the rest of the images formed the training data. The network was trained using mini-batches of 100 images for a total of 10 epochs, where an epoch is a full pass through all of the images in the training set. At intervals during training the network is presented with the test set of images. For each test image the output from the network is saved as the reconstructed image. The network does not learn during the test phase, so the test images are new to the network each time and are not contained in the training set.

3 Results

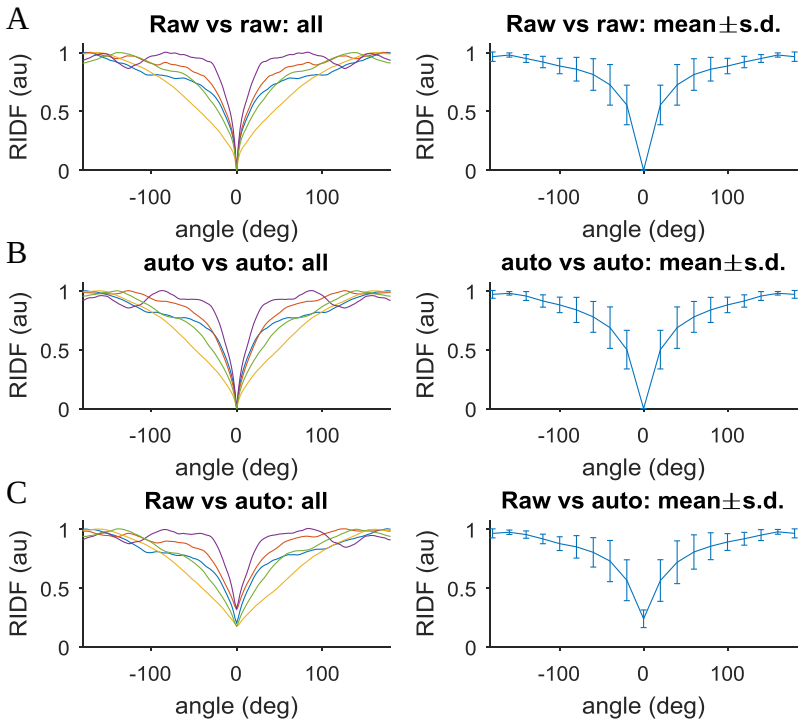

Fig. 3. Rotational image difference function (RIDF) plots showing **A**. original image vs original image, **B**. reconstructed image vs reconstructed image, **C**. original image vs reconstructed image. Data are shown for five test images, with images compared to the same image at all possible rotations. Left column shows individual RIDFs for each image. Right column shows mean with standard deviation error bars.

In order to see if navigationally useful information is retained once an image is passed through the autoencoder, we use the RIDF metric to see if reconstructed images can be used to regain a heading. To do this, we take 5 new images, which are separate from the training and test images, and look at the RIDFs that are produced when these images are compared with rotated versions of themselves. This simulates rotation of the direction that a robot is facing, as if it were turning on the spot through 360º, to find the best-matching heading to its memory.

As a control condition, we first examine RIDFs from raw images. As can be seen in Fig. 3A, the pixel difference is lowest at the centre where there is no rotation, and increases rapidly as the current viewing direction is rotated away from its original orientation. This results in a characteristic V shape, indicating the image can be used

to recover heading information from nearby positions. There is some variability in the widths of the V, which to some extent reflects the area over which the image can be used to regain a heading (broader V is indicative of a wider region). More importantly, the V is relatively smooth, especially towards the centre, which is good as the presence of multiple, deep, minima indicate there may be a problem with visual aliasing (that is, one location being confused with another).

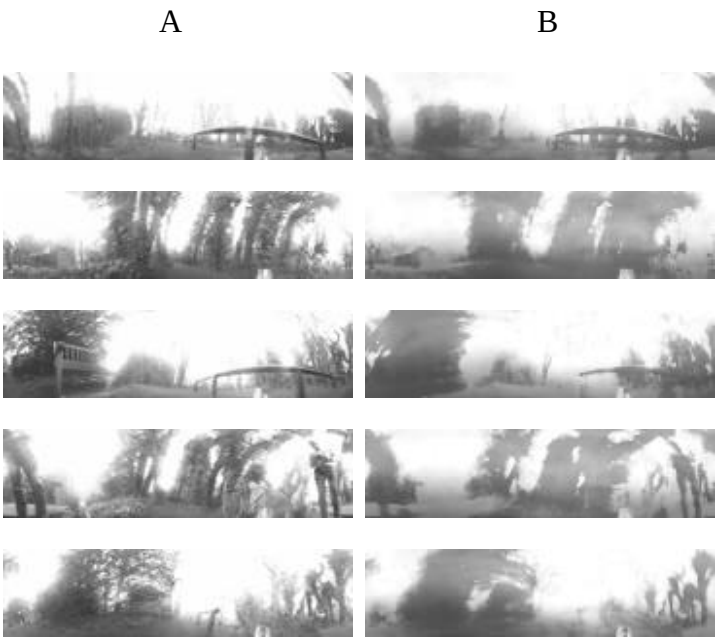

Fig. 4. Input and output images from the autoencoder. **A**. Original images supplied as test input to the network **B**. Reconstructed images from output of the deep autoencoder

We next compare the images that have been reconstructed by the autoencoder with themselves to see whether navigationally useful information has been retained (Fig. 3B). This is indeed the case and the RIDFs are very similar to those for the raw images and if anything, perhaps a little more consistent near the true heading. This bodes well for using such an autoencoder as the visual front-end of a robot. Further, if we compare the raw images with the reconstructed images, while as expected the minimum value is not zero, the correct heading is achieved, the RIDF shapes look very similar to the other two cases (Fig. 3C). This indicates that similar information is retained by the encoded network as is in the raw image. The implications are that an agent's perceptual system and memory system can work with different encodings, perhaps with perceptual input being minimally processed and memories being processed to a higher degree.

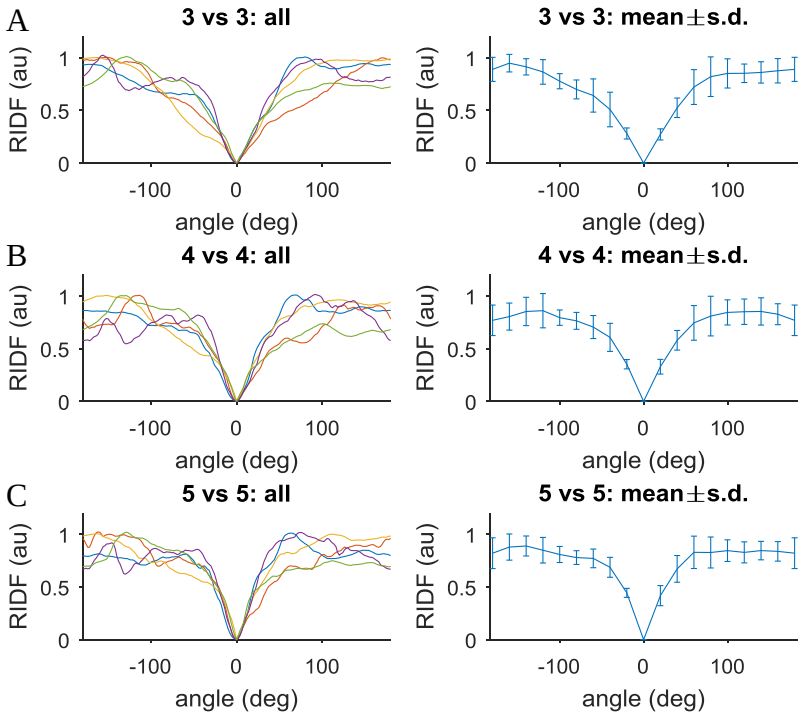

Fig. 5. Rotational image difference function (RIDF) plots derived from the outputs of the three smallest layers of the autoencoder network: **A**. layer 3, 512 neurons; **B**. layer 4, 128 neurons; **C**. layer 5, 64 neurons. Data are shown for five test images, with outputs compared to outputs for the same image at all possible rotations. Left column shows individual RIDFs for each image. Right column shows mean with standard deviation error bars.

The reason for this can be seen if we compare the images and output after training (Fig. 4). Much of the high spatial frequency content is removed, leaving what looks like a low-resolution image. This is interesting as it was recently shown in simulation that navigation could be better with images whose resolution is on the order of degrees, similar to the ant's eye, than with high-resolution images [10]. It is thus perhaps no surprise that navigation is achievable with the reconstructed images, though the fact that one can use the raw image as a comparator further suggests that high spatial frequency information is somewhat redundant for this task.

While we have shown that the reconstructed images can be used for navigation, and thus that the compression in the central layers does not remove this information, the reconstructed images are of the same dimension as the originals. Using them instead

of the originals does not therefore present a saving in terms of either memory (which scales with N, the number of pixels) or the computation needed to derive a heading from them (which scales roughly as N^2 depending on the method used). We next therefore assess the RIDFs which results from the three smallest layers of the encoder network, the 3rd 4th and 5th layers which have 512, 128 and 64 units respectively. These are show in Fig 5 A, B and C respectively.

Despite the over 10-fold reduction in dimension, the RIDFs remain and have smooth Vs around the correct heading (Fig. 5) even for the smallest layer (Fig. 5C). The RIDFs contain more spurious optima in headings away from the correct one than the reconstructed images (compare individual RIDFs, left column, in Fig. 5 at azimuths over 90 degrees from the centre with Fig. 3A,B, left column), suggesting visual aliasing could be a greater problem. However, the width of the Vs is if anything slightly greater than the reconstructed images. This is curious as the trend within the layers is for a slightly greater width for layer 3 (Fig. 5A) than 4 or 5 (Fig. 5B,C, respectively) perhaps suggesting there is an optimal size for image compression between raw and very low-resolution, again echoing [10].

4 Conclusion

In summary, we have shown that we can use a deep autoencoder to derive a compressed representation of natural images that preserves the information required to derive a heading. Further, the information persists at different levels of abstraction within the network and does not require the decoder part of the network. This work thus has implications for robotic navigation as well as understanding the insect visual pathway.

In terms of robotics, this work has implications for the visual processing that could be performed in our route navigation algorithms. our route navigation algorithm proceeds by training a neural network to encode a route [19] either by training a classifier to classify if an image is part of a route or not [20], or by learning the familiarity of the training images [7,21]. Currently, aside from lowering the resolution and basic image normalization, we do not pre-process the images. This work suggests that if we pre-trained a deep autoencoder, we could use the initial layers to provide a compact representation of the images. We could then train these compact representations online to learn specific routes, which would result in much more efficient networks with encodings tuned to regularities in the natural habitat.

Turning to the implications for biology, at this stage, more analysis needs to be done to assess what features of the images are being encoded by the networks and whether these features could be plausibly extracted in the different layers of processing in the insect visual pathway. For instance, as the network is fully connected, the central layer encodings, which appear at least sufficient for homing, could be combining information from across the visual field. This hints at intriguing parallels with the

wide-field integration which occurs after the initial two or three stages of visual processing in insect visual pathway, which will be borne out or disproved by analysis of the connectivity. In addition, due to the rather parallel nature of the first stage of visual pathways, we will additionally examine convolutional networks to assess what filters are learned as preliminary work produces similar results to the fully connected network presented here.

Acknowledgements

This work was funded by the Newton Agri-Tech Program RICE PADDY project (no. STDA00732). AP and PG were also funded by EPSRC grant EP/P006094/1.

References

1. Graham, P., Philippides A: Insect-Inspired Vision and Visually Guided Behavior. In Bhushan B and Winbigler H D (eds.) Encyclopedia of Nanotechnology, Springer (2015)
2. Wehner, R., Räber, F.: Visual Spatial memory in desert ants, Cataglyphis bicolor. Experientia, 35, 1569-1571 (1979)
3. Cartwright, B.A., Collett, T.S.: Landmark Learning in Bees - Experiments and Models. Journal of Comparative Physiology, 151, 521-543 (1979)
4. Wehner, R: Desert ant navigation: How miniature brains solve complex tasks. Karl von Frisch lecture. J Comp Physiol A 189, 579-588 (2003)
5. Wehner, R.: The architecture of the desert ant's navigational toolkit (Hymenoptera: Formicidae). Myrmecol News 12, 85-96 (2009)
6. Smith, L., Philippides, A., Graham, P., Baddeley, B., Husbands, P.: Linked local navigation for visual route guidance. Adapt Behav 15, 257–271 (2007)
7. Baddeley, B., Graham, P., Husbands, P., Philippides, A.: A Model of Ant Route Navigation Driven by Scene Familiarity. PLoS Comput Biol 8(1), e1002336 (2012)
8. Ardin, P., Peng, F., Mangan, M., Lagogiannis, K., Webb, B.: Using an insect mushroom body circuit to encode route memory in complex natural environments. PLoS Comput Biol, 12(2), .e1004683 (2016)
9. Möller, R., Vardy, A.: Local visual homing by matched-filter descent in image distances. Biol Cybern 95: 413-430 (2006)
10. Wystrach, A., Dewar, A., Philippides, A., Graham, P.: How do field of view and resolution affect the information content of panoramic scenes for visual navigation? A computational investigation. J Comp Physiol A, 202, 87-95 (2016)
11. Dewar, A., Wystrach, A., Graham, P., Philippides, A.: Navigation-specific neural coding in the visual system of Drosophila. Biosystems, 136, 120-127 (2015)
12. Philippides, A., Baddeley, B., Cheng, K., Graham, P.: How might ants use panoramic views for route navigation? J Exp Biol 214, 445-451 (2011)
13. LeCun, Y., Bengio, Y., Hinton, G.: Deep learning. Nature, 521(7553), pp.436–444. (2015)
14. Zeil. J., Hofmann, M., Chahl, J.: Catchment areas of panoramic snapshots in outdoor scenes. J Opt Soc Am A 20, 450-469 (2003)
15. Stürzl, W., Zeil, J.: Depth, contrast and view-based homing in outdoor scenes. Biol Cybern 96, 519-531 (2007)

16. Hinton, G. E., & Salakhutdinov, R. R.: Reducing the dimensionality of data with neural networks. Science, 313(5786), 504-507 (2006)
17. Bengio Y., Courville A. C., Vincent P.: Unsupervised feature learning and deep learning: A review and new perspectives. CoRR, abs/1206.5538 (2012)
18. Kingma, D., Ba, J.: Adam: A method for stochastic optimization. arXiv:1412.6980. (2014)
19. Philippides, A., Graham, P., Baddeley, B., Husbands, P.: Using neural networks to understand the information that guides behavior: a case study in visual navigation. Artificial Neural Networks, 227-244 (2015)
20. Baddeley, B., Graham, P., Philippides, A., Husbands, P.: Holistic visual encoding of ant-like routes: Navigation without waypoints. Adapt Behav, 19, 3-15, (2011)
21. Baddeley, B., Graham, P., Philippides, A., Husbands, P.: Models of visually guided routes in ants: Embodiment simplifies route acquisition. In International Conference on Intelligent Robotics and Applications, 75-84. Springer Berlin Heidelberg, (2011)